

REMARKS

Applicant respectfully requests the continued examination, the reconsideration of the present application and the consideration of the following remarks. This amendment is submitted in response to the Final Office Action mailed on March 15, 2005. Claims 1-19 were rejected. Claims 1, 7, 10 and 17 have been amended. No new matter has been added.

Claims 10-12, 14 and 16 were rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,101,510 (hereinafter "Stone") in view of U.S. Patent No. 5,761,673 (hereinafter "Bookman"). Claims 1-9, 13 and 17-19 were rejected under 35 U.S.C. 103(a) as being unpatentable over Stone in view of Bookman and "Understanding ActiveX and OLE" (hereinafter "Chappell"). Claim 15 was rejected under 35 U.S.C. 103(a) as being unpatentable over Stone in view of Bookman and further in view of "ActiveX Programming Unleashed" (hereinafter "Chen"). As discussed below, the pending claims are patentable over the above references.

Stone discloses a web browser control that allows application program developers to incorporate web browser functionality into application programs. The web browser control exposes web-browsing functionality to application programs through an application program interface. This interface comprises member functions, events and properties. The member functions provide high level services such as Navigate to a URL, go forward or backward in a navigation stack, or refresh the display of an HTML page. The events are notification messages that the web browser control sends to a host application to notify the application about actions that have taken place or are about to take place. The properties provide status information about an instance of a control.

Bookman shows a web server that can generate dynamic web pages.

Chen has only a general description of JavaScript objects.

Chappell shows a method to implement "linked or embedded data", such as Excel data, in another application, such as Word, using the OLE technology.

Independent Claim 1

Contrary to the presently claimed invention of claim 1, Stone does not teach or suggest having a server that is capable of communicating with the non-network based application to access an object of the non-network based application in response to a request from the network based application. Stone does not show or suggest that browser control server (42) is capable of accessing the objects of the application (44) which uses the browser control server (42) to incorporate the web browser functionality into the application. The cited references, such as Bookman, Chappell and Chen, also do not show or suggest such a feature as recited in the independent claim 1. Thus, even if the references were combined in a way indicated in the Office Action, the combination does not teach or suggest at least the features of the presently claimed invention that are included in the following language of claim 1:

“... wherein the server is capable of communicating with the non-network based application to access an object of the non-network based application in response to a request from the network based application.”

This language is supported by the present specification (e.g., see, page 6, paragraph [017]; page 12, paragraph [030]; etc.).

Further, the Office Action relied upon Chappell to provide a motivation to construct a combination including a non-network based application.

The Examiner asserted that “it would be obvious to combine the teachings of STONE with the teachings of BOOKMAN and CHAPPELL in order to provide a link and embedded data from a server without being aware of what kind of application the other is” (Page 7, the second paragraph, Office Action mailed March 15, 2005). However, these references lack

any *motivation* for such a combination. Further, Chappell does not teach or suggest providing “a link”.

Note that the original description of Chappell (page 174) is:

“Because the interfaces are entirely generic, a container need never know what kind of servers are providing linked or embedded data for its documents. ... for instance, Word knows only that it’s interacting with some server that follows OLE’s rules for servers; it doesn’t know, or care, that the server is Excel.” (page 174, Chappell).

From this description of Chappell, a person skilled in the art understands that Chappell is **not** showing the motivation to provide a link (**hyperlink**) in the container such as Word. The example of Chappell is to embed data, such as Excel data, in Word. Thus, it is not proper to assert that Chappell provides the motivation “to provide a link”. The description of Chappell is about “linked or embedded data”, such as Excel data.

In page 174 of Chappell, the statement of “Neither is aware of what kind of application the other is” relates to the standard container interfaces and the standard server interfaces. The interfaces are standard such that neither a container nor a server is aware of what kind of application the other is. Thus, if Word is implemented as a container and Excel is implemented as a server, neither Word nor Excel is aware of what kind of application the other is from the standard interfaces. The example of Chappell illustrates combining Word with Excel using OLE. Clearly, neither Stone nor Bookman nor Chappell shows the motivation of incorporating web browser functionality into Word or Excel.

Thus, at least for the above reasons, the rejection for claim 1 in view of Chappell, is improper.

Independent Claims 7, 10 and 17

Language similar to that discussed above for claim 1 is also included in independent claims 7, 10 and 17. Thus, the present invention as claimed in claims 1, 7, 10 and 17, and their corresponding dependent claims, is patentable over the cited references, even if they were combined in a way suggested in the Office Action.

Claim 15

Claim 15 recites “wherein the object is a JavaScript object”. Note that the base claim 10 of claim 15 recites “a request identifying an object ... from a third party application ...”.

Contrary to the presently claimed invention, the host application of a browser control of Stone can only get the web page from the browser control to “add HTML code or scripting language to the page and then invoke the hypertext viewer to render and display the code” (see, e.g., Col. 4, lines 23-29; Col. 23, lines 47-58, Stone). Stone does not show that, once the script language is rendered to generate an object, the host application can *identify* the object in a request to access the datum in the object generated from the script language. Chen has only a general description of JavaScript objects.

Thus, even if Stone, Bookman and Chen were combined, in the combination the host application would at most add a script in JavaScript to the web page and then ask the browser control to render it. There is no indication, suggestion or motivation for the host application of Stone to further send a request identifying the JavaScript object and to access the datum included in the JavaScript object. Thus, even if Stone, Bookman and Chen were combined in a way as suggested in the Office Action, the combination does not show the features corresponding to what is recited in the claim.

The remaining claims depend from at least one of the independent claims discussed above, and therefore include at least some of the distinguishing claim limitations as discussed above. As a result, the remaining claims are also patentable.

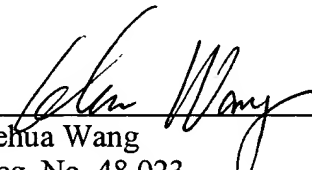
Applicant respectfully submits that the pending claims are patentable over the cited references. If the Examiner believes a telephone conference would expedite or assist in the allowance of the present application, the Examiner is invited to call at (408) 720-8300.

Authorization is hereby given to charge our Deposit Account No. 02-2666 for any charges that may be due or credit any overages.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 7/26, 2005



Lehua Wang
Reg. No. 48,023

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025-1026
(408) 720-8300